# What is transforms in CSS?

By modifying the coordinate space, CSS transforms change the shape and position of the affected content without disrupting the normal document flow. This guide provides an introduction to using transforms.

CSS transforms are implemented using a set of CSS properties that let you apply affine linear transformations to HTML elements. These transformations include rotation, skewing, scaling, and translation both in the plane and in the 3D space.

**Important Point:-** Only elements positioned by the box model can be transformed. As a rule of thumb, an element is positioned by the box model if it has display: block.

# CSS transforms properties

Two major properties are used to define CSS transforms: transform and transform-origin

1. **transform-origin :**
   Specifies the position of the origin. By default, it is at the center of the element and can be moved. It is used by several transforms, like rotations, scaling or skewing, that need a specific point as a parameter.
2. **transform :**
   Specifies the transforms to apply to the element. It is a space-separated list of transforms, which are applied one after the other, as requested by the composition operation. Composite transforms are effectively applied in order from right to left.

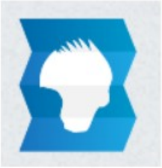
# Example:-

Here is an unaltered image of the MDN logo:

## Rotating

Here is the MDN logo rotated 90 degrees from its bottom-left corner.

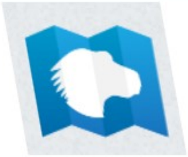
```
<img style="transform: rotate(90deg);
           transform-origin: bottom left;"
     src="logo.png">
```



## Skewing and translating

Here is the MDN logo, skewed by 10 degrees and translated by 150 pixels on the X-axis.

```
<img style="transform: skewx(10deg) translatex(150px);
           transform-origin: bottom left;"
     src="logo.png">
```



# What is transition in CSS?

CSS transitions provide a way to control animation speed when changing CSS properties. Instead of having property changes take effect immediately, you can cause the changes in a property to take place over a period of time. For example, if you change the color of an element from white to black, usually the change is instantaneous. With CSS transitions enabled, changes occur at time intervals that follow an acceleration curve, all of which can be customized.

## Defining transitions

CSS Transitions are controlled using the shorthand transition property. This is the best way to configure transitions, as it makes it easier to avoid out of sync parameters, which can be very frustrating to have to spend lots of time debugging in CSS.

You can control the individual components of the transition with the following sub-properties:

1. transition-property
2. transition-duration
3. transition-timing-function
4. transition-delay

The shorthand CSS syntax is written as follows:

```
div {
    transition: <property> <duration> <timing-function> <delay>;
}
```

or we can use these properties as:- (Its your choice to use shortand or not)

```
a {
  color: #fff;
  background-color: #333;
  transition: all 1s ease-out;
}

a:hover,
a:focus {
  color: #333;
  background-color: #fff;
}
```

## Let us see the propety by a example :-

In this example we have a box and when user hover mouse on the box it start to rotate and the color of the box also changes. You can try it by copying below code and running it.

```
.box {
    border-style: solid;
    border-width: 1px;
    display: block;
    width: 100px;
    height: 100px;
    background-color: #0000FF;
    transition: width 2s, height 2s, background-color 2s, transform 2s;
}

.box:hover {
    background-color: #FFCCCC;
    width: 200px;
    height: 200px;
    transform: rotate(180deg);
}
```